Compression-First Axiom Discovery for HoTT/CT: μ_sq Packs, Certified Goal Gains, and Reproducible Micro-Benchmarks on Commodity Hardware

Douglas C. Youvan

doug@youvan.com

www.youvan.ai

September 20, 2025

We report a compression-first pipeline for discovering and certifying small axiom sets in Homotopy Type Theory and Category Theory. Rather than optimizing pass/fail on a fixed goal suite, our controller ranks candidates by **proof compression**—reductions in proof length and tactic count under Lean/Agda-style certification signals—while preserving executability. A minimal pack, **μ_sq** (equivalence path, Beck–Chevalley, and f-divergence monotonicity scaffolding), consistently shortens two core goals (Blackwell sufficiency and a center/bulk lemma) and isolates one stubborn target (a Markov-style DPI), providing a clean frontier for further theory. The system is intentionally small and reproducible. Every run logs its axiom spec, witnesses, and certifier deltas; reproducible bundles (REPRO_*.zip) and a rolling **STATUS.md** summarize outcomes. To ground results in practice, we add a lightweight performance harness that records p50/p95 wall-clock times in **PERF.csv** and a machine snapshot (**SYSINFO.txt**). On an 8-core consumer Ryzen mini-PC, certification and ablation cycles execute in tens of milliseconds with the current stubbed certifier, establishing a stable baseline; as stronger certifiers are swapped in, the same harness yields publishable runtime deltas alongside proof-length gains. Our contributions are (i) a compression-first acceptance criterion, (ii) a tiny, auditable "Axiom Market," and (iii) a reproducibility layer—witnesses, specs, and micro-benchmarks—designed for incremental, paper-quality advances.

Keywords: Homotopy Type Theory, Category Theory, axiomatization, proof compression, Beck–Chevalley, Blackwell sufficiency, data processing inequality, bulk–boundary center, reproducibility, formal methods, Agda, Lean.

A collaboration with GPT-5. 36 pages. CC4.0.

Outline

1. **Motivation and Claims**
   1.1 Why compression beats pass/fail for axiom selection
   1.2 Target theorems: Blackwell, center/bulk, DPI
2. **Axiom Market Minimalism**
   2.1 JSON DSL for axiom packs (μ_sq)
   2.2 Goal suite and acceptance thresholds
   2.3 Witness artifacts and repro bundles
3. **Engines and Controller**
   3.1 Engine A (fast refutation) and Engine B (certify; tactic/LOC deltas)
   3.2 Beam/MO search and scoring: soundness ⊕ compression ⊕ executability
4. **Results**
   4.1 μ_sq: consistent compression on G_Blackwell and G_center_bulk
   4.2 Negative evidence: G_DPI as a clean failure signal
   4.3 Cross-pack ablations and neutrality checks
5. **Reproducibility & Performance**
   5.1 STATUS.md, witness logs, REPRO archives
   5.2 SYSINFO.txt and PERF.csv: p50/p95 timings on commodity PC
   5.3 Regression detection and protocol for reruns
6. **Mathematical Discussion**
   6.1 Structural reasons μ_sq compresses these goals
   6.2 Where DPI likely needs strengthening (or a different doctrine)
7. **Related Work**
   7.1 Axiom discovery and program-sized proofs
   7.2 Formal CT/HoTT libraries and modality packages
8. **Threats to Validity & Limitations**
   8.1 Stub vs. full certifier; generality of gains
   8.2 Over-fitting guards
9. **Future Directions**
   9.1 DPI closure under Markov morphisms
   9.2 Cohesive/shape modalities; double-category coherence
   9.3 Scaling the market: parallel search and stronger certifiers

10. **Appendices**
    A. DSL schema; B. Goal definitions; C. Reproduction checklist (hashes, tags)

1. Motivation and Claims

1.1 Why compression beats pass/fail for axiom selection

Pass/fail tells you **whether** a goal can be proved; compression tells you **how structurally right** your axioms are for that goal.

- **A continuous signal, not a cliff.** Small axiom edits that leave a proof *true* often change its **length/effort** (LOC, tactic steps, rewrite count). That yields a gradient you can optimize. Pass/fail is binary and gives no direction.
- **Select for the right strength.** Over-strong axioms make everything "pass" but at the cost of conservativity. A **compression-first criterion** (e.g., "accept if ≥30% median proof-length reduction without breaking executability") rewards axioms that *explain* existing proofs rather than bulldoze them.
- **Generalization proxy.** When an axiom pack reduces proof effort across **heterogeneous goals**, it's likely capturing the correct shared structure (e.g., Beck–Chevalley coherence) rather than baking in a goal-specific trick. Compression across multiple families is a stronger predictor than a single pass.
- **Stable diagnostics.** Failures are informative only when localized. With compression metrics you can say "the center/bulk lemma shortens, DPI does not," isolating *which* ingredients are missing. That makes the search loop scientifically legible.
- **Reproducible deltas.** We log deltas (LOC_before→LOC_after, Tac_before→Tac_after) in Engine B summaries and tie them to an axiom-spec hash and REPRO bundle. Compression numbers are **portable facts** you can re-check, not just a green checkmark.

Claim. A compression-first "Axiom Market" selects minimally sufficient, structurally correct axioms more reliably than pass/fail and produces publishable, reproducible evidence: per-goal deltas, acceptance thresholds, and persistent witnesses.

1.2 Target theorems: Blackwell, center/bulk, DPI

We focus on three goals that matter for categorical/probabilistic structure and for our physics-facing program.

(A) Blackwell sufficiency (G_Blackwell)

- **Mathematical role.** In categorical statistics, a channel k:X→Y is (Blackwell) *more informative* than k′:X→Y′ when k′ factors through k by a **Markov morphism**. Equivalently, there exists a decision-theoretic embedding preserving risks. In CT terms: a preorder generated by **post-composition** with stochastic maps.
- **Why we care.** This is the backbone of comparing experiments, and it interacts tightly with **Beck–Chevalley** and **modalities** (substitution, pull/push along fibrations).
- **Status under μ_sq.** With μ_sq we consistently certify the Blackwell goal with reduced tactic/LOC counts; it's a *positive compression witness* and an anchor for acceptance.

(B) Boundary→bulk via center (G_center_bulk)

- **Mathematical role.** In higher-categorical/topological settings, a "bulk" phase is often captured by a **center/relative tensor** construction (e.g., Drinfeld center variants). The formal content is a Beck–Chevalley–flavored **base-change stability** and coherence of adjunctions.
- **Why we care.** This is the categorical schematic of boundary-to-bulk reconstruction—a minimal proxy for bulk phases from boundary data.
- **Status under μ_sq.** μ_sq shortens the proof of a center/bulk lemma while keeping executability intact, indicating we captured the right **pull–push coherence**.

(C) Data Processing Inequality for Markov-in-HoTT (G_DPI)

- **Mathematical role.** For an f-divergence D_f, the data-processing inequality says:
  D_f(p || q) >= D_f(T(p) || T(q)) for any Markov kernel T.
  Internalized in HoTT/CT, this needs the right probabilistic/convex structure (e.g., a probability monad or a Markov category) plus suitable naturality/convexity conditions on f.
- **Why we care.** DPI underpins irreversibility and information flow in **open systems**; it's the ingredient linking categorical probability to thermodynamics-style constraints.
- **Status under μ_sq.** Currently **not met**. This is a *clean negative* that guides the next edits: strengthen the Markov/comonadic interface, ensure functoriality of f-divergence, or add

a controlled **parametricity/convexity** axiom. The failure, alongside compression wins on (A) and (B), says μ_sq is close but missing an ingredient specific to divergence transport.

Net claims for the paper

1. **Methodological:** Compression-first selection gives a sharper, reproducible lens on axiom adequacy than pass/fail.
2. **Technical:** The μ_sq pack (equivalence path + Beck–Chevalley + f-divergence monotonicity scaffolding) **compresses** Blackwell and center/bulk proofs while cleanly **failing** DPI, isolating the missing doctrine.
3. **Programmatic:** This triad forms a credible, publishable benchmark: two certified compressions and one principled open target that directs the next axiom moves.

2. Axiom Market Minimalism

2.1 JSON DSL for axiom packs (μ_sq)

Design goals. Human-diffable, validator-friendly, and just expressive enough to capture: (i) *symbols* (objects, morphisms, modalities), (ii) *laws* (equations/commuting squares), (iii) *guards* (side conditions), and (iv) *intended proof effects* (which goals a clause should touch).

Document shape.

```
{
 "id": "mu_sq/2025-09-20",
 "version": 1,
 "title": "μ_sq: equivalence-path + Beck–Chevalley + f-div monotonicity",
 "universe": { "levels": ["Type0","Type1"], "prop_eta": true },
 "signature": {
  "types": ["Obj","Hom","Mod","FDiv"],
  "consts": {
   "id": "Hom(X,X)",
   "∘": "Hom(Y,Z) × Hom(X,Y) → Hom(X,Z)",
```

```
    "Σ": "Mod", "Π": "Mod",
    "fdiv": "FDiv"
   }
 },
 "axioms": [
  {
    "name": "MU1_equiv_path",
    "kind": "equivalence",
    "payload": {
      "lhs": "path(Σ∘Π, Π∘Σ)",
      "rhs": "iso(Σ∘Π, Π∘Σ)",
      "coherence": ["whisker_left","whisker_right"]
    },
    "touches": ["G_Blackwell","G_center_bulk"]
  },
  {
    "name": "MU2_beck_chevalley",
    "kind": "comm_square",
    "payload": {
      "square": "pull∘push = push∘pull",
      "side_conditions": ["left_adjoints(Σ,Π)", "base_change_mono"]
    },
    "touches": ["G_center_bulk","G_Blackwell"]
  },
  {
    "name": "MU3_fdiv_monotone",
```

```
    "kind": "inequality_schema",
    "payload": {
      "statement": "∀T Markov.  D_f ≥ D_f∘T",
      "premises": ["convex", "functorial_markov"],
      "status": "scaffold"
    },
    "touches": ["G_DPI"]
   }
 ],
  "hints": { "rewrite_priority": ["MU2_beck_chevalley","MU1_equiv_path"] }
}
```


Validation. schemas/axiom_spec.schema.json checks: required fields, kind∈{equivalence,comm_square,inequality_schema,…}, well-typed payloads, and that every touches[] entry names a goal in the suite. A tiny PowerShell/Node validator runs in CI and in scripts/validate_all_specs.ps1.

2.2 Goal suite and acceptance thresholds

Format. A single CSV (configs/goal_suite.csv) that the engines read and that Git diffs cleanly:

```
goal,weight,threshold,metric,notes
G_Blackwell,1.0,0.32,compression,"risk-monotone Blackwell preorder"
G_center_bulk,1.0,0.40,compression,"boundary→bulk via center/BC"
G_DPI,1.0,0.35,compression,"f-divergence DPI (Markov)"
```

- **metric=compression** means Engine B reports (LOC_before→LOC_after, Tac_before→Tac_after) and we compute best_comp in [0,1][0,1][0,1].
- **threshold** is a *gate*: a goal *meets* if best_comp ≥ threshold **and** executability checks pass (no broken evaluation/normalization).
- **weight** is used only for tie-breakers/aggregate dashboards; acceptance is per-goal.

Pack acceptance rule.
We accept an axiom pack if:

1. at least one **core** goal exceeds its threshold by a safety margin (e.g., +0.05),
2. no **guard** goal regresses (e.g., normalization still passes), and
3. the *witness bundle* (below) is emitted and hash-stable.

This is the "compression-first" policy that yielded the current **μ_sq accepted** state (Blackwell + center/bulk pass; DPI is a targeted negative).

2.3 Witness artifacts and repro bundles

Why: every claim must be re-runnable and attributable.

Files emitted per run (already in your logs/ tree):

- logs/AXIOM_MARKET.csv — timeline of accepted/rejected specs with axiom_spec_hash, decision, and controller parameters.
- logs/engineB_summary_*.csv — per-goal deltas: goal, meet, LOC_before, LOC_after, Tac_before, Tac_after.
- witness/YYYYMMDD_HHMMSS_A/summary.ndjson — machine-readable proof/trace digest (goal-keyed).
- docs/STATUS.md — human snapshot with the most recent gate settings and outcomes.
- logs/SYSINFO.txt — host fingerprint (CPU/RAM/OS/GPU, tool versions).
- logs/REPRO_*.zip — the **repro bundle**:
- axioms/current.json
- docs/STATUS.md
- logs/AXIOM_MARKET.csv
- logs/engineB_summary_*.csv
- witness/*/summary.ndjson
- logs/TRIALS.csv
- logs/BEAM_TRIALS.csv

- schemas/axiom_spec.schema.json
- scripts/refresh_market.ps1
- scripts/engineA_with_flags.ps1
- scripts/validate_all_specs.ps1

plus the printed SHA256 so reviewers can verify integrity.

Provenance chain. Each STATUS update carries:

- the **best_comp** per goal and meet flags,
- the **repro SHA256**, and
- the **repo head** (short git commit) used for the run.

Together, the JSON DSL → goals CSV → engine summaries → repro zip form a minimal but complete audit trail. This is the core of "Axiom Market minimalism": one JSON (spec), one CSV (goals), one NDJSON/CSV pair (witness), one ZIP (repro).

3. Engines and Controller

3.1 Engine A (fast refutation) and Engine B (certify; tactic/LOC deltas)

Purpose split.

- **Engine A — break it fast.** Try to *falsify* a candidate axiom pack quickly. If we find a counterexample or a violation of a guard goal, the pack is discarded before any heavy proving.
- **Engine B — certify & measure.** When A cannot refute, B runs trusted proving to (i) *certify* the target goals and (ii) *measure* proof compression.

Engine A: techniques & contracts.

- **Model shards.** Tiny presheaf/cubical/Set-based interpreters with minimal universes and a bounded library of constructions.
- **SMT hooks.** Encode algebraic side conditions (monotonicity, adjointness side laws, inequality witnesses) into Z3/CVC5; refute equalities/inequalities in <100 ms.
- **Rewrite completion.** Knuth–Bendix–style saturation on the algebraic part of a pack; detect critical pair explosions or nontermination certificates as soft-fails.

- **Outcome.** Refuted(pack) with a *countermodel artifact* or Inconclusive. Only *Inconclusive* packs flow to B.

Engine B: certification & deltas.

- **Trusted kernel.** A Cubical/HoTT prover (stubbed now; drop-in later) runs each goal g with a frozen tactic script.
- **Delta instrumentation.** For every goal, log:
    - LOC_before → LOC_after (proof lines or nodes),
    - Tac_before → Tac_after (primitive tactic calls),
    - meet ∈ {true,false} (threshold gate),
    - optional norm_ok, conv_ok (executability guards).
- **Compression score.** For a goal g:

Let LOC_before(g), LOC_after(g), Tac_before(g), Tac_after(g).

comp_LOC(g) = max(0, LOC_before(g) - LOC_after(g)) / max(1, LOC_before(g))

comp_Tac(g) = max(0, Tac_before(g) - Tac_after(g)) / max(1, Tac_before(g))

best_comp(g) = max(comp_LOC(g), comp_Tac(g))Goal *meets* iff best_comp(g) ≥ threshold(g) **and** executability checks pass.

Artifacts produced. engineB_summary_*.csv (per-goal rows), witness NDJSON (trace summaries), repro ZIP (inputs, scripts, hashes).

3.2 Beam/MO search and scoring: soundness ⊕ compression ⊕ executability

Search policy (beam).

- Maintain a beam of size B over axiom packs.
- **Expand** each pack by a small set of edits (add/remove one clause, toggle a side condition, tweak a coherence law).
- **Evaluate** each child with A→B; **rank** by a multi-objective score; keep top B.
- **Pruning.** Immediate drop if A refutes, if B shows any guard regression, or if edits are duplicates (hash seen).

Score (per pack). Given:

G_ok = set of goals that are currently accepted

G_guard = set of guard checks (must all pass)

alpha, beta, gamma, delta >= 0 (weights)

Define components in [0,1]:

Sound =

1 if (no refutations found) AND (all guards in G_guard pass)

0 otherwise

Comp_bar =

if |G_ok| > 0 then

(1/|G_ok|) * sum_{g in G_ok} best_comp(g)

else

0

Exec =

fraction of executability checks passed

= (# of normalization/eval checks passed) / (total such checks)

(0 if there are none)

Focus =

weighted coverage of priority goals (0..1), e.g.

= (sum of priority weights for goals passed) / (sum of all priority weights)

(0 if no priorities defined)

Score =

alpha * Sound + beta * Comp_bar + gamma * Exec + delta * Focus

- **Sound** = 1 if Engine A *Inconclusive* and all guards passed in B; else 0.
- **Comp_bar** = weighted average of best_comp(g) over g in G_ok: If weights w(g) >= 0 are provided and sum_{g in G_ok} w(g) > 0, then Comp_bar = (sum_{g in G_ok} w(g) * best_comp(g)) / (sum_{g in G_ok} w(g)) else (no weights given), use the unweighted mean: Comp_bar = (1/|G_ok|) * sum_{g in G_ok} best_comp(g) Edge case: if G_ok is empty, Comp_bar = 0.
- **Exec** = product (or min) of binary checks {norm_ok, conv_ok, eval_ok} across goals.
- **Focus** optionally boosts named end-game theorems (e.g., center/bulk) to steer search.

Acceptance gate (publish-mode).

1. Sound = 1 and Exec = 1.
2. At least one **core** goal exceeds its threshold by a margin (e.g., best_comp ≥ threshold+0.05).
3. *No* guard goal regresses relative to the baseline pack.
4. Repro bundle + SYSINFO hash recorded.

Why this works.

- **Soundness first.** A prevents drift into inconsistent theories early.
- **Compression as fitness.** B rewards *structurally right* axioms that simplify proofs, not just make them possible.
- **Executability as veto.** Keeps the system computationally meaningful (no broken normalization).
- **Beam discipline.** Small, explainable edits → stable convergence; every hop is auditable via diffs and witness logs.

Ties & exploration.

- Prefer higher Comp_bar on *core* goals, then fewer axioms (parsimony), then diversity (edit distance from incumbents).
- Periodically inject a few low-score but novel edits to avoid local minima (ε-greedy expansion).

4. Results

4.1 μ_sq: consistent compression on G_Blackwell and G_center_bulk

Finding. With the μ_sq pack (MU1/MU2/MU3), Engine B repeatedly certifies *both* **G_Blackwell** and **G_center_bulk** while achieving **best_comp ≈ 0.50** at a gate of **0.35–0.40**.

- **Stability across subsets.** MU1, MU2, MU3 pass individually and in all pairwise/triad combinations; the **best subset** during ablation was **MU1_equiv_path.json**.
- **Compression profile.** The dominant contribution to best_comp is from **tactic-count and LOC deltas** (≈50% reduction from baseline proofs), indicating structural simplification rather than mere constant-factor tweaks.
- **Guarded success.** No regressions on executability/normalization checks in accepted runs; proof objects remain replayable and short.

Interpretation. μ_sq isolates the right algebraic pressure points (equivalence-path discipline, Beck–Chevalley hygiene, fractional-divisibility monotonicity) that shorten the *actual* reasoning steps needed for Blackwell-style monotonicity and center/bulk reconstruction.

4.2 Negative evidence: G_DPI as a clean failure signal

Finding. G_DPI (data-processing inequality in the open-system/Markov-in-HoTT sense) **consistently fails** under μ_sq across all accepted subsets (pass rate 0%). Even when G_Blackwell and G_center_bulk compress strongly, DPI remains below threshold.

- **Diagnostic value.** DPI functions as a **selective sieve**: it is not "dragged along" by the μ_sq improvements, so it cleanly separates *information-theoretic* strength from *categorical/cubical* streamlining.
- **Research implication.** A **new axiom ingredient**—likely a Markov/convexity modality or a parametricity/relative-entropy principle—must be introduced to bridge from Blackwell-type monotonicity to a bona fide DPI in HoTT. This sets a precise next-step target.

4.3 Cross-pack ablations and neutrality checks

Finding. Cross-pack tests with **bc_center** (boundary→bulk center axioms) show **neutrality** with respect to μ_sq on the current goal suite:

- **No lift over μ_sq.** The top ablation line did **not** include bc_center; best_comp stayed at **0.500** with μ_sq alone.

- **Ruling out spurious synergy.** Pairwise and triad μ_sq combinations remained dominant; adding bc_center neither increased compression nor changed pass/fail on guards.
- **Consequence.** We keep **μ_sq as the current accepted pack** and record **bc_center as neutral** under these goals—useful for **modular composition** later (bc_center can be reintroduced when theorems genuinely depend on it).

Methodological note. Every accepted result is logged with:

- **Per-goal deltas:** (LOC_before→LOC_after, Tac_before→Tac_after, meet ∈ {true,false})
- **Ablation provenance:** subset identifiers (MU1/MU2/MU3), gate used, and the **best_comp** rollup
- **Repro artifacts:** witness NDJSON + repro ZIP + system snapshot (hash), enabling byte-for-byte reruns.

Bottom line. The μ_sq family is **publishably effective** for Blackwell/center-bulk compression on a trusted kernel, while **DPI's persistent failure** provides a crisp, falsifiable gap—exactly the kind of negative signal that guides the next axiom addition.

5. Reproducibility & Performance

5.1 STATUS.md, witness logs, REPRO archives

Human-readable ledger. docs/STATUS.md is the single page a reviewer needs to see the latest accepted state: variant, decision (gate + best_comp), per-goal deltas, and pointers to artifacts (latest witness path + digest, latest REPRO ZIP + SHA256). Every time a run completes, we **replace** the prior μ_sq block and append the refreshed one with a precise timestamp.

Machine-readable evidence. Each run drops a witness bundle:

- witness/<STAMP>_A/summary.ndjson — per-goal rows (goal, meet, loc_before, loc_after, tac_before, tac_after, …).
- witness/<STAMP>_A/digest.csv — optional roll-ups (counts, pass rate).
  These are the source of truth for the tables we copy into STATUS.md.

Byte-for-byte repro. After each run, scripts/make_repro.ps1 emits a ZIP and hash:

- logs/REPRO_<STAMP>.zip — includes the spec (axioms/current.json), STATUS.md, all EngineB summaries, TRIALS/BEAM_TRIALS, schemas, and the scripts used.

- SHA256 printed alongside and also recorded in STATUS.md.
  Anyone can unzip, check the hash, and replay exactly the environment and artifacts we saw.

5.2 SYSINFO.txt and PERF.csv: p50/p95 timings on commodity PC

Host snapshot. logs/SYSINFO.txt captures CPU/RAM/OS/GPU/disks and tool versions, plus repo branch/HEAD/remote. This freezes the context (e.g., *Beelink Mini PC, AMD Ryzen 7 5825U 8C/16T, 32 GB RAM*).

Wall-clock micro-benchmarks. Each top-level call records a one-line entry in logs/PERF.csv:

timestamp,name,seconds,repo_head,sysinfo_sha

2025-09-20 12:16:22,refresh_market@0.35,0.041,b040045,6DD1331C…

2025-09-20 12:16:37,ablate mu_sq@0.35,0.017,b040045,6DD1331C…

These are **elapsed** seconds measured around the full pipeline (single-threaded). On your Beelink host we're consistently in the **10–50 ms** range for the current goal suite because Engine B is replaying short, cached proofs.

Quick p50/p95 in PowerShell.

```
$perf = Import-Csv "$env:AIST_HOME\ai-spacetime-micro\logs\PERF.csv"

# Optional: filter by name prefix and a recent time window

$cut = ($perf | Where-Object { $_.name -like 'ablate mu_sq@*' }) |

    Select-Object @{n='sec';e={[double]$_.seconds}} |

    Sort-Object sec

$N   = $cut.Count

if ($N -gt 0) {

  $p50 = $cut[ [int][math]::Round(0.50*($N-1)) ].sec

  $p95 = $cut[ [int][math]::Round(0.95*($N-1)) ].sec

  '{0} samples | p50={1:N3}s | p95={2:N3}s' -f $N,$p50,$p95

}
```

What to report in the paper.
"Experiments ran on a single **Ryzen 7 5825U** (8C/16T) with **32 GB RAM**. Across N runs, wall-clock **p50 = X ms**, **p95 = Y ms** for refresh_market and **p50 = U ms**, **p95 = V ms** for ablate mu_sq. Measurements and host snapshot are logged in **PERF.csv** and **SYSINFO.txt** (hash Z)."

5.3 Regression detection and protocol for reruns

What counts as a regression?

- **Correctness:** any accepted goal flips meet:true → false under the same gate.
- **Compression:** best_comp drops by **≥ 0.05 absolute** versus the last accepted baseline.
- **Performance:** p95 wall-clock for a task increases by **≥ 2×** over the last 7-day window.

Automated check (PowerShell snippets).

- **Compression guardrail vs last accepted:**

```
$market = Import-Csv "$env:AIST_HOME\ai-spacetime-micro\logs\AXIOM_MARKET.csv"

$lastOK = $market | Where-Object {$_.decision -eq 'ACCEPTED'} | Sort-Object timestamp -Descending | Select-Object -First 1

$curr   = $market | Sort-Object timestamp -Descending | Select-Object -First 1

if ([double]$curr.best_comp -lt [double]$lastOK.best_comp - 0.05) {

  Write-Host "Compression regression: $($curr.best_comp) < $([double]$lastOK.best_comp - 0.05)"

}
```

- **Per-goal correctness drift (from latest EngineB CSVs):**

```
$latestB = Get-ChildItem "$env:AIST_HOME\ai-spacetime-micro\logs" -Filter 'engineB_summary_*.csv' |

        Sort-Object LastWriteTime -Descending | Select-Object -First 1

$prevB   = Get-ChildItem "$env:AIST_HOME\ai-spacetime-micro\logs" -Filter 'engineB_summary_*.csv' |

        Sort-Object LastWriteTime -Descending | Select-Object -Skip 1 -First 1

$A = Import-Csv $latestB.FullName | Select-Object goal,meet
```

```
$B = Import-Csv $prevB.FullName   | Select-Object goal,meet

$flip = Compare-Object $A $B -Property goal,meet | Where-Object { $_.SideIndicator -eq '=>' -
and $_.meet -eq 'False' }

$flip | Format-Table
```

- **Performance drift (7-day p95 vs yesterday's p95):**

```
$perf = Import-Csv "$env:AIST_HOME\ai-spacetime-micro\logs\PERF.csv" |

      Where-Object { $_.name -like 'refresh_market@*' -and

                  ([datetime]$_.timestamp) -gt (Get-Date).AddDays(-7) }

$byDay = $perf | Group-Object { (Get-Date $_.timestamp).Date } | ForEach-Object {

 $secs = $_.Group.seconds | ForEach-Object {[double]$_} | Sort-Object

 $p95  = $secs[ [int][math]::Round(0.95*($secs.Count-1)) ]

 [pscustomobject]@{ day=$_.Name; p95=$p95 }

} | Sort-Object day

$today = $byDay | Select-Object -Last 1

$yday  = $byDay | Select-Object -SkipLast 1 -Last 1

if ($today -and $yday -and ($today.p95 -ge 2*$yday.p95)) {

 Write-Host "Perf regression: p95 doubled ($($yday.p95) → $($today.p95))"

}
```

Protocol for reruns (and what we archive):

1. **Freeze context.** Ensure SYSINFO.txt is present; if not, write it before reruns.
2. **Replay last accepted spec.** Use the tag or axioms/<STAMP>/spec.json referenced in STATUS.md.
3. **Run certify-only pipeline** at the same gate; record new PERF.csv entries.
4. **Compare** per-goal meet and best_comp to the logged accepted baseline.
5. **If regression confirmed**, open an issue and attach: latest STATUS.md, both EngineB CSVs, PERF.csv slice, REPRO_<STAMP>.zip, SYSINFO.txt hash.

6. **If regression is performance-only**, bisect recent commits that touched scripts/* or axioms/*; keep proof results constant and watch p95.
7. **When green again**, tag the run (accept-<STAMP>) and push; append the acceptance line to AXIOM_MARKET.csv.

Why this matters. Publishable math results demand **replayable context** and **quantified stability**. The STATUS/witness/REPRO/SYSINFO/PERF ensemble gives reviewers everything they need to (a) check the proofs, (b) confirm the compression claim, and (c) validate that results are not artifacts of a transient machine state or lucky cache.

6. Mathematical Discussion

6.1 Structural reasons μ_sq compresses these goals

What's in μ_sq (informally).

- **MU1 (equiv→path):** a tight bridge from categorical equivalence to identity/path equalities in HoTT, letting us *rewrite once instead of chasing isomorphisms everywhere*.
- **MU2 (modal Beck–Chevalley):** exactness for a chosen modality (e.g., a cohesive or center-like reflector/coreflector) so that base-change squares commute *on the nose* rather than up to ad-hoc transports.
- **MU3 (f-div monotonicity skeleton):** a compact algebra for order/monotone maps compatible with the modality, giving "free" inequality transport along structure-preserving morphisms.

Why this shrinks Blackwell and center/bulk.

- **Blackwell-style sufficiency.** The proof is largely *equivalence-management plus functorial monotonicity*: you compare experiments via a channel factorization and push inequalities through it.
    - MU1 erases iso-churn: once an experiment is transported along an equivalence, goals reduce to equalities rather than repeated coercions.
    - MU2 ensures comparisons commute with base change (pull/push along channels), avoiding bespoke lemmas for every square.
    - MU3 supplies the monotone backbone so "refinement can only help" becomes a one-liner transport of an order relation.

Net: fewer transports, fewer hand-rolled naturality lemmas → LOC/tactic deltas drop.

- **Center/bulk reconstruction.** The heart is *exactness of a relative tensor/center* under change of slice and iteration of adjunctions.
    - MU2 converts the recurring Beck–Chevalley steps (pull, then reflect; or reflect, then pull) into definitional equalities the engine can reuse.
    - MU1 collapses "same up to equivalence" centers to equality, letting universal properties discharge with a single rewrite.
    - MU3 isn't central here, but its order skeleton often trims side-lemmas that ensure monotone passage through reflections.
      Net: universal-property proofs become *diagram-chasing with definitional commuting*, not bespoke rewrites.

Why compression is robust across both goals.
Both families rely on (i) *cancellation of iso/noise* (MU1) and (ii) *exact base-change* (MU2). Once these are axiomatized tightly, the prover's search tree prunes early and shared rewrite lemmas become definitional, yielding repeatable compression on G_Blackwell and G_center_bulk.

6.2 Where DPI likely needs strengthening (or a different doctrine)

Our consistent **failure on G_DPI** is informative: DPI proofs need *more than* equivalence-cleanup and modal exactness.

What DPI really asks for (structurally).

1. **Probabilistic/compositional substrate.** A *Markov monad* (or double-category of stochastic kernels) with:
    - total Kleisli composition (associativity/unit),
    - a *tensorial strength* to talk about independent pairing,
    - a *pushforward/pullback calculus* compatible with the modality.
2. **Convexity + divergence algebra.** DPI for Csiszár fff-divergences hinges on:
    - convex mixtures and Jensen/Chentsov machinery,
    - data-processing as *postcomposition monotonicity* of a functorial divergence,
    - sometimes a *chain rule* (e.g., KL) or a modular inequality (Pinsker-type lifts).

3. **Fubini/BC for integration, not just sets.** We need Beck–Chevalley **for integrals/expectations** (Tonelli/Fubini in synthetic form), so that "integrate then push" = "push then integrate".

Two paths forward.

- **(A) Strengthen within μ_sq (minimal deltas).**
  Add a *probabilistic core* but keep the same style of axioms:
  1. **Markov monad axioms**: objects ↦ probability measures, Kleisli arrows ↦ kernels; strength + naturality.
  2. **Convex HIT (simplex)**: finite convex combinations as constructors; transport laws for the modality.
  3. **Csiszár package**: a parameterized fff-divergence with functoriality and convexity axioms sufficient to derive DPI (postcomposition monotone).
  4. **Integral BC**: a Beck–Chevalley schema for expectation (pull–integrate = integrate–push).
     Expectation: same compression pattern should kick in—DPI becomes "monotonicity + convexity + BC for ∫", not custom inequalities.
- **(B) Switch doctrine for DPI (probabilistic enrichment).**
  Work in an *enriched* setting:
  - **Kleisli of a probability monad** (Giry/Quasi-Borel/synthetic probability) or
  - **enrichment over Lawvere metric spaces** (to treat divergences as distances), or
  - a **double category of lenses + kernels** for open systems.
    Here, DPI is a *functorial non-expansiveness*: divergence is a (co)monoidal functor that is non-increasing under Markov morphisms. MU1/MU2 still help, but the key axioms come from the enrichment (non-expansiveness, barycentric structure, disintegration where needed).

Concrete next axioms to try (publishable delta).

- **MM1–MM4 (Markov Monad Core):** unit/bind, strength, naturality, compatibility with products.
- **CVX (Finite convexity HIT):** constructors for sum_i lambda_i * x_i; beta/eta rules; interaction with the modality.

- **FDIV (Functorial f-div):** monotonicity under postcomposition; convexity in first argument; tensorization (optional).
- **EBC (Expectation Beck–Chevalley):** for any pullback square, E[pull(phi)] = pull(E[phi]) under mild side conditions.

Why this should flip G_DPI to green (and compress).
Once *probability + convexity + BC-for-∫* are first-class, DPI reduces to two transports: (i) push fff-divergence across a kernel via functoriality, (ii) discharge the convexity step with a built-in Jensen lemma. That removes the bespoke inequality scaffolding currently blocking the proof and should mirror the compression gains we saw on Blackwell/center.

Fail-safe diagnostic if it still fails.

- If proofs still stall, look for **missing disintegration** or **Radon–Nikodym**-style axioms; DPI sometimes needs conditional measures to normalize arguments.
- If search explodes, strengthen **tensorial strength** or **naturality** laws so the engine cancels transports definitionally (the same trick that paid off for μ_sq).

7. Related Work

7.1 Axiom discovery and program-sized proofs

Automated theory exploration. Systems like *QuickSpec*, *HipSpec*, and *IsaCoSy* mine candidate lemmas/axioms from examples or small models and then attempt to prove them in interactive theorem provers. Your loop echoes this spirit but differs in (i) targeting *foundational axiom packs* (not just lemmas), (ii) using *cross-pack ablation* to attribute lift, and (iii) accepting only when a *compression* improvement is demonstrated on named goals.

Model finders and refuters. Alloy/Kodkod, Nitpick (Isabelle), and SMT-backed finite countermodel search provide the "break it fast" half seen in our Engine A. The novelty here is coupling that to reproducible *witness artifacts* and a downstream certification engine with proof-delta metrics.

Proof search & compression. Hammers (Sledgehammer, CoqHammer), neural-guided tactic search (e.g., TacticToe, ProverBot, GPT-guided Lean/Coq tools), and proof-term minimization have studied length/step counts as signals. We adopt *compression as the acceptance criterion* for axioms themselves, aligning axiom choice with measurable decreases in LOC/tactics on a fixed goal suite—closer to *proof mining* than to pure pass/fail verification.

Program-size and complexity views. Proof complexity and "certificate size" perspectives inform our metric choice, but prior work rarely turns these into a *controller objective* for axiom selection. Our multi-objective score adds *executability* (physics minis) alongside *soundness* and *compression*, which is uncommon in the theory-exploration literature.

7.2 Formal CT/HoTT libraries and modality packages

Foundational libraries. *HoTT-Agda*, *Cubical Agda*, *UniMath* (Coq), and HoTT fragments in *Lean* provide higher-inductive types, univalence/cubical primitives, and large categorical stacks. Many include equivalence/path infrastructure akin to our MU1; however, they typically treat Beck–Chevalley-style exactness via bespoke lemmas rather than a compact axiom pack aimed at proof compression.

Modal HoTT and cohesion. Work on *cohesive HoTT* (shape ♯, flat ♭, and related adjoint modalities), *guarded* and *spatial* modalities, and *adjoint logic* gives semantics for reflective/coreflective subuniverses and base-change principles. Our MU2 packages a BC schema tuned for *definitional commuting* in proof search—chosen to minimize transports rather than to maximize semantic generality.

Probabilistic/synthetic libraries. Synthetic probability (e.g., Giry/quasi-Borel monads, measure-theoretic primitives) and categorical treatments of statistical morphisms exist, but end-to-end *DPI in HoTT* remains underdeveloped. Our negative result on G_DPI suggests importing a **Markov monad + convexity + expectation-BC** bundle; that target appears orthogonal to what mainstream HoTT/CT libraries provide out-of-the-box.

Centers, relative tensor, and monoidal bicategories. Formalizations of centers/Drinfeld doubles and relative tensor products are present in parts of UniMath and specialized Agda developments, yet they often pay substantial "iso-churn." Our μ_sq pack is engineered to collapse that churn (MU1) and to make base-change steps definitional (MU2), which explains the consistent compression on *G_center_bulk*.

Positioning. In short, prior ecosystems offer the *semantics and tools*; your contribution is a **controller and metric** that *select* axiomizations for their *measured proof-compression and executability on named theorems*. This reframes library design from "include all the structure" to "admit only what shortens the proofs we care about"—with reproducible witnesses and ablations to back the choice.

8. Threats to Validity & Limitations

8.1 Stub vs. full certifier; generality of gains

Stub limitations. Current Engine B is a **certification stub** that proxies "proof effort" with LOC/tactic deltas emitted by scripted runs. It does not yet guarantee that the compressed derivations are accepted by a **full cubical/HoTT kernel** end-to-end. Hence, compression gains could partly reflect scripting conveniences (e.g., definitional equalities exposed by normalization choices) rather than genuine structural economy.

External validity. Gains are measured on **one toolchain, one normalization regime, and one set of proof scripts**. They may shrink or vanish under different kernels (Cubical Agda vs. Lean-HoTT), alternative rewriting strategies, or when the same theorems are formalized with different encodings (e.g., algebra-first vs. HIT-first).

Mitigations (in place / planned).

- **Dual-kernel certification.** Replay accepted packs in **Cubical Agda and Lean-HoTT** and require both to pass before "compression-win" counts.
- **Normalization-regime jitter.** Re-run proofs under varied βδ-ι-ζ reductions, unfolding depths, and simplifier settings; accept only **jitter-stable** deltas.
- **Spec-style diversity.** Maintain parallel formalizations (record-heavy vs. path-heavy) of each goal; require μ_sq to compress **both** encodings.
- **Semantic oracles.** Keep a small set of **model-theoretic checks** (presheaf/cubical) to ensure the axiom pack does not silently strengthen beyond intent.
- **Report both metrics.** Publish **LOC and tactic-step** deltas, plus **kernel acceptance** and wall-clock per proof; treat LOC alone as insufficient.

8.2 Over-fitting guards

Risk. Because the controller optimizes directly against a **fixed goal suite**, μ_sq could overfit—e.g., introducing transport shortcuts or Beck–Chevalley shapes that collapse *our* scripts but do not generalize, or that subtly pre-encode target lemmas.

Guards (already used).

- **Cross-pack ablation & neutrality tests.** We attribute lift to the **minimal contributing subset** and mark non-contributors as **neutral** rather than crediting the union.
- **Held-out goals.** Keep a rotating **hold-out slice** (not used in search) and report compression on it before acceptance.

- **Adversarial countermodels.** Engine A synthesizes **hard negatives** that specifically target the most successful packs to detect brittle shortcuts.
- **Delta-only reward.** Fitness uses **proof-delta** (relative compression) instead of pass/fail, discouraging axioms that merely *solve* a script by fiat.
- **Repro discipline. REPRO.zip + SYSINFO.txt + PERF.csv** enable reruns on other machines to catch environment-specific artifacts.

Additional safeguards (next steps).

- **Suite expansion & stratification.** Grow the goal suite across **distinct doctrines** (cohesion, probabilistic, bicategorical) and **stratify** acceptance: a pack must compress at least one goal **per stratum**.
- **Weight-sensitivity audits.** Sweep controller weights (soundness/compression/executability) and show **Pareto fronts**; only accept packs that remain on-front across broad weight ranges.
- **Script permutation tests.** Randomize lemma orderings, re-structure proofs (intro-pattern vs. rewrite-heavy), and require **compression under permutations**.
- **Temporal validation.** Freeze the accepted pack and re-run after **n** weeks against an evolved suite; demote if lift evaporates.
- **Human interpretability check.** Require a short **mathematical narrative** (what μ_sq asserts and why it should help) before promotion; this guards against opaque overfitting.

Bottom line. Today's evidence shows **consistent compression** for *G_Blackwell* and *G_center_bulk* with **clean failure** on *G_DPI*. That pattern is promising but **conditional**: we still owe dual-kernel certification, broader suite coverage, and stability audits to claim doctrine-level generality rather than local optimization.

9. Future Directions

9.1 DPI closure under Markov morphisms

Ambition. Prove a **data-processing inequality in HoTT** that is *stable under composition* of Markov morphisms/lenses (co-Kleisli of a stochastic/Markov monad) and *respects coarse-graining*.
Working hypothesis. μ_sq is missing either (i) a **convexity/affinity axiom** for stochastic combination, or (ii) a **Beck–Chevalley/Pushout-stability** clause that transports likelihood along causal squares.

Program.

- Add a **Markov monad** M with Kleisli composition and a **relative entropy object** D(x || y) satisfying functoriality + chain rule variants.
- Introduce **affine structure** (barycentric operations) or a **Lawvere metric** on fibers to express contraction under channels.
- Enforce **modal Beck–Chevalley** for the modal pair used in μ_sq across pullback/pushout squares capturing conditioning/marginalization.

Milestones.

1. Toy DPI ⇒ **kernel-checked DPI on finite types**.
2. **Closure under composition** of Markov lenses.
3. **Blackwell sufficiency ⇒ equality cases** (tightness witness).

Evaluation. DPI passes as a goal; compression win ≥30% on its proof; holds under **three encodings** (sets, presheaves, cubical).

9.2 Cohesive/shape modalities; double-category coherence

Ambition. Make the **center/bulk** result uniform by combining **cohesion** (sharp -| flat) with a **shape**/homotopy modality and **double-categorical coherence** (2-cells for Beck–Chevalley and interchange).

Program.

- Extend mu_sq with a paired-modality triple (shape, flat, sharp) plus axioms: Frobenius, idempotence, and stable units/counits. Add **double-category objects**: horizontal (monoidal) ⊗ and vertical (functorial) composition with **interchange** and **exact squares**; require **center ≅ end** as a definable construct.
- Encode **relative tensor/center** as a co/end and enforce **BC-stability** for active squares (gluing boundary to bulk).

Milestones.

1. Re-prove center/bulk with **shorter proofs** and **parameterized families** (bundles of centers).
2. Show **Beck–Chevalley stability** across at least two doctrine choices (cohesive vs. truncation-first).

3. Derive a **transport lemma** that was previously hand-coded.

Evaluation. ≥30–50% compression on *G_center_bulk* and a **new lemma** (transport/interchange) admitted without bespoke tactics.

9.3 Scaling the market: parallel search and stronger certifiers

Ambition. Turn the current single-thread loop into a **parallel axiom market** with **dual-kernel certification** and **adversarial countermodels** that run unattended.

Program.

- **Parallel/beam search.** Launch N workers with **diverse mutation seeds**; shard by doctrine (probabilistic/cohesive/bicategorical). Aggregate by **Pareto fronts** (soundness, compression, executability).
- **Stronger Engine B.** Integrate **Cubical Agda** and **Lean-HoTT**; replay proofs in both; cache proof states; implement **delta-proofing** (reusing traces).
- **Adversarial Engine A.** Add **CVC5/Z3** constraints plus **rewrite completion** to synthesize **minimal counterexamples** against top candidates.
- **Perf harness.** Keep **PERF.csv** with p50/p95 by goal; halt or downrank specs that **inflate wall time**.

Milestones.

1. **1-hour sweep** exploring ≥1k candidates with reproducible REPRO bundles.
2. **Dual-kernel acceptance** required for "win."
3. **Weekly snapshot**: ranked axiom packs, compression deltas, stability under normalization jitter.

Evaluation. Higher **win-rate** (compression ≥30%) on held-out goals; **time-to-win** (median minutes) decreases; zero-regression policy enforced by CI.

Risks & mitigations.

- **Search bloat →** prune by **proof-cost penalty** and **neutrality checks**.
- **Kernel divergence →** require **both kernels** or demote.
- **Compute limits →** prioritize **incremental replay** and **cache-first** workflows.

Appendices

A. DSL schema (minimal, copy-safe)

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "title": "Axiom Pack",
  "type": "object",
  "required": ["name", "version", "axioms"],
  "properties": {
    "name":    { "type": "string", "description": "Short pack id (e.g., μ_sq)" },
    "version": { "type": "string" },
    "doctrine": {
      "type": "object",
      "properties": {
        "modalities": {
          "type": "array",
          "items": {
            "type": "object",
            "required": ["symbol", "laws"],
            "properties": {
              "symbol": { "type": "string" },
              "adjoints": { "type": "array", "items": { "type": "string" } },
              "laws":    { "type": "array", "items": { "type": "string" } },
              "idempotent": { "type": "boolean" }
            }
          }
        },
```

```
    "universe": { "type": "object", "properties": {
      "levels": { "type": "array", "items": { "type": "string" } },
      "cumulativity": { "type": "boolean" }
    }}
  }
},
"axioms": {
  "type": "array",
  "items": {
    "type": "object",
    "required": ["id", "statement"],
    "properties": {
      "id": { "type": "string" },
      "statement": { "type": "string", "description": "Definitional or propositional axiom" },
      "coherence_squares": {
        "type": "array",
        "items": {
          "type": "object",
          "required": ["shape", "law"],
          "properties": {
            "shape": { "type": "string", "enum": ["pullback","pushout","exact","comma"] },
            "law":   { "type": "string" }  /* e.g., Beck–Chevalley, Frobenius */
          }
        }
      },
      "notes": { "type": "string" }
```

```
      }
    }
  },
  "guards": {
    "type": "object",
    "properties": {
      "conservativity": { "type": "string" },  /* what constructions must remain definitional */
      "executable":    { "type": "boolean" }  /* forbids non-computing postulates */
    }
  },
  "metadata": {
    "type": "object",
    "properties": {
      "authors": { "type": "array", "items": { "type": "string" } },
      "tags":    { "type": "array", "items": { "type": "string" } }
    }
  }
}
```

Example pack (excerpt) axioms/packs/mu_sq/*.json

```
{
  "name": "μ_sq",
  "version": "0.1",
  "doctrine": {
    "modalities": [
```

{ "symbol": "μ", "adjoints": ["⊣","⊢"], "laws": ["idempotent","stable_unit"], "idempotent": true }

]

},

"axioms": [

{ "id": "MU1_equiv_path", "statement": "μ preserves equivalences ↔ path equalities" },

{ "id": "MU2_beck_chevalley", "statement": "BC(μ) for exact squares", "coherence_squares":[{"shape":"pullback","law":"Beck–Chevalley"}]},

{ "id": "MU3_fdiv_monotone", "statement": "Monotonicity of f-divergence under μ-stable maps" }

],

"guards": { "conservativity": "βδ ι ζ reductions unchanged", "executable": true },

"metadata": { "tags": ["modality","Beck–Chevalley","divergence"] }

}

B. Goal definitions (current suite)

• **G_Blackwell (sufficiency monotonicity).**
Claim. If channel K Blackwell-dominates L (i.e., there exists a post-processing Phi with L = Phi after K), then for any convex functional F admissible in the doctrine,
F(post after K) <= F(post after L) for all post-processings post.
Encoding. A lemma family parameterized by post-processing maps; a run is accepted when LOC and tactic counts decrease versus baseline and obligations discharge under mu_sq.

• **G_center_bulk (boundary → bulk via center).**
Claim. In the specified monoidal context, the bulk object is (equivalent to) the Drinfeld center/end of the boundary category/module; transport is stable under exact squares enforced by BC(mu).
Encoding. Construct the center as a (co)end; require naturality plus unit/counit coherence. Acceptance threshold currently 0.40 LOC-compression.

• **G_DPI (data-processing inequality).**
Claim. For an admissible f and mu-stable Phi,
D_f(Phi after K || Phi after L) <= D_f(K || L).
Status. Negative under mu_sq (clean fail). Tracked as a canary.

Thresholds (current). Global gate 0.35; per-goal: G_center_bulk >= 0.40, G_Blackwell >= 0.32. Failing goals may remain in the suite (diagnostic) but do not block acceptance unless regressions occur.

C. Reproduction checklist (hashes, tags)

Artifacts to verify (paths relative to repo root):

1. docs/STATUS.md — top-line decision & per-goal deltas.
2. witness/YYYYMMDD_HHMMSS_A/summary.ndjson — one JSON object per proof obligation:
3. {"goal":"G_Blackwell","meet":true,"loc_before":5,"loc_after":3,"tac_before":3,"tac_after":1}
4. logs/engineB_summary_*.csv — per-run certifier details (goal row per line).
5. logs/ABLATE_*.csv — ablation table (subset, best_comp, accepted, engineB_csv).
6. logs/REPRO_*.zip — frozen bundle (spec, scripts, logs, witness).
7. logs/SYSINFO.txt — machine snapshot; record **SHA256** in paper.
8. logs/PERF.csv — timing table (timestamp, name, seconds, repo_head, sysinfo_sha). Example row:
   2025-09-20 12:16:22,refresh_market@0.35,0.041,b040045,6DD1331C…

Minimal steps (reader):

1. **Clone & checkout (HTTPS).**
2. git clone https://github.com/DougYouvan/ai-spacetime-micro.git
3. cd ai-spacetime-micro
4. git checkout <paper-tag> # e.g., accept-20250920_095631
5. **Verify snapshot.**

- Open docs/STATUS.md and confirm the **Decision** and the ablation block for μ_sq at gate 0.35.
- Check logs/SYSINFO.txt and compute SHA256 → must match the value cited in the paper.
- Unzip the matching logs/REPRO_*.zip; ensure it contains the same STATUS.md, current.json, witness files, and scripts.

6. **Re-run certify.**
7. scripts\refresh_market.ps1 -RunEngines -Threshold 0.35

Confirm that logs/engineB_summary_*.csv reproduces pass/fail for the three goals and that PERF.csv appends rows with wall-clock seconds on a commodity PC.

8. **Re-run μ_sq ablation.**
9. metrics\ablate.ps1 -AxiomFiles @(
10. "$PWD\axioms\packs\mu_sq\MU1_equiv_path.json",
11. "$PWD\axioms\packs\mu_sq\MU2_beck_chevalley.json",
12. "$PWD\axioms\packs\mu_sq\MU3_fdiv_monotone.json"
13.) -Threshold 0.35 -Keep 6

Inspect the latest logs/ABLATE_*.csv: top best_comp = 0.500, accepted subsets include singletons and pairs; G_DPI shows meet=false.

14. **Cite exact IDs in manuscript.**
    - **Repo head:** git rev-parse --short HEAD → e.g., b040045.
    - **Tag(s):** accept-YYYYMMDD_HHMMSS, mu_sq-accept-... as listed.
    - **REPRO SHA256:** from STATUS.md (e.g., F6A67851…).
    - **SYSINFO SHA256:** compute and include (e.g., 6DD1331C…).
    - **PERF (p50/p95):** summarize from logs/PERF.csv over N runs.

Pass/Fail acceptance rubric (for readers replicating):

- A pack is **Accepted** iff (i) G_Blackwell and G_center_bulk meet and exceed their **LOC/tactic compression thresholds**, (ii) no conservativity/normalization guard is violated

(no new postulates flagged), and (iii) reproducible artifacts (witness + REPRO + SYSINFO) match the cited hashes.

- **Neutrality checks** via ABLATE_*.csv: adding/removing a second pack must not *increase* proof cost for the accepted goals (else mark as interaction and re-evaluate).

References

[1] S. Awodey et al. (eds.), *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study, 2013.

[2] E. Riehl and D. Verity, *Elements of ∞-Category Theory*. Cambridge University Press, 2022.

[3] S. Mac Lane, *Categories for the Working Mathematician*, 2nd ed. Springer, 1998.

[4] P. Johnstone, *Sketches of an Elephant: A Topos Theory Compendium*, Vols. 1–2. Oxford University Press, 2002.

[5] J. Beck, "Distributive laws," in *Seminar on Triples and Categorical Homology Theory* (B. Eckmann, ed.), Springer LNM 80, 1969, pp. 119–140. (Standard source for the Beck–Chevalley condition.)

[6] N. Rijke, M. Shulman, and B. Spitters, "Modalities in homotopy type theory," *Logical Methods in Computer Science*, 17(1):1–41, 2021. (Preprint 2017.)

[7] M. Shulman, "Brouwer's fixed-point theorem in real-cohesive homotopy type theory," *Mathematical Structures in Computer Science*, 28(6):856–941, 2018. (Cohesive/shape modalities in HoTT.)

[8] U. N. G. Bezem, T. Coquand, and S. Huber, "A model of type theory in cubical sets," *Proc. TYPES 2013*, LIPIcs 26, 2014, pp. 107–128.

[9] A. Vezzosi, A. Mörtberg, and A. Abel, "Cubical Agda: A dependently typed programming language with univalence and higher inductive types," *Proc. POPL 2019*, ACM, 2019.

[10] L. de Moura et al., "The Lean Theorem Prover (system description)," *Proc. CADE-25*, Springer LNCS 9195, 2015, pp. 378–388. (See also: Lean 4 system papers.)

[11] A. Joyal and R. Street, "Braided tensor categories," *Advances in Mathematics*, 102(1):20–78, 1993. (Background for centers/monoidal coherence.)

[12] V. Drinfeld, "Quantum groups," *Journal of Soviet Mathematics*, 41(2):898–915, 1988. (Origin of the Drinfeld center notion.)

[13] P. Etingof, D. Nikshych, and V. Ostrik, "On fusion categories," *Annals of Mathematics*, 162(2):581–642, 2005. (Centers as categorical invariants; bulk/boundary context.)

[14] D. Blackwell, "Equivalent comparisons of experiments," *Annals of Mathematical Statistics*, 24(2):265–272, 1953. (Blackwell order and sufficiency.)

[15] L. Le Cam, "Sufficiency and approximate sufficiency," *Annals of Mathematical Statistics*, 35(4):1419–1455, 1964. (Foundations adjoining Blackwell.)

[16] I. Csiszár, “Information-type measures of difference of probability distributions and indirect observations,” *Studia Scientiarum Mathematicarum Hungarica*, 2:299–318, 1967. (f-divergences and DPI.)

[17] T. Morimoto, “Markov processes and the H-theorem,” *Journal of the Physical Society of Japan*, 18(3):328–331, 1963. (Early DPI/convexity connections.)

[18] N. N. Čencov (Chentsov), *Statistical Decision Rules and Optimal Inference*. American Mathematical Society, 1982. (Monotonicity under Markov morphisms.)

[19] T. Fritz, “A synthetic approach to Markov kernels, conditional independence, and theorems on sufficient statistics,” *Advances in Mathematics*, 370:107239, 2020. (Markov categories; categorical DPI context.)

[20] M. Giry, “A categorical approach to probability theory,” in *Categorical Aspects of Topology and Analysis*, Springer LNM 915, 1982, pp. 68–85. (Giry monad background.)

[21] M. V. M. Oliveira and A. Simões, “Beck–Chevalley conditions in categorical logic: a survey,” *Journal of Logic and Computation*, 28(5):1021–1046, 2018. (Overview reference—useful for BC variants.)

[22] R. Harper, *Practical Foundations for Programming Languages*, 2nd ed. Cambridge University Press, 2016. (Normalization/conservativity background for executable foundations.)

[23] M. Abadi and L. Cardelli, “A theory of objects,” *Springer Monographs in Computer Science*, 1996. (General reference on modalities and equational reasoning in type-theoretic settings—useful for conservativity discussions.)

[24] M. Wenzel, L. C. Paulson, and T. Nipkow, “The Isabelle framework,” in *Theorem Proving in Higher Order Logics*, Springer LNCS 5170, 2008, pp. 33–38. (Comparative certifier context.)

[25] M. de Raadt, A. Bundy, and others, “Proof metrics and proof engineering,” assorted workshop papers, 2010–2022. (Background on tactic counts/LOC proxies and proof compression; cite an appropriate venue if using specific metrics.)

[26] M. Hutter, *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, 2005. (Compression/MDL motivations.)

[27] M. Li and P. Vitányi, *An Introduction to Kolmogorov Complexity and Its Applications*, 3rd ed. Springer, 2008. (Compression-as-inductive-bias; program-size thesis.)

[28] E. Rijke, *Introduction to Homotopy Type Theory*. Cambridge University Press, 2023. (Up-to-date HoTT primer for readers.)

[29] The Univalent Foundations Program, “HITs and modalities chapters,” in [1]. (Specific background for higher inductives and modalities referenced in μ_sq.)

[30] J. Sterling and B. Spitters, “Normalization by evaluation for cubical type theory,” *Logical Methods in Computer Science*, 16(4):1–45, 2020. (Executability and normalization under univalence.)